Actividad grupal. Desarrollo de un problema de planificación

Resumen

Introducción

Definición de planificación

Según la RAE se le define a la planificación como aquel plan metódicamente organizado y de gran amplitud para la obtención de un objetivo determinado. (Asale, s. f.) Por otro lado, la planificación en IA es una de sus áreas encargadas de, mediante programación automática, diseñar planes de acción que serán ejecutadas por un agente inteligente para la obtención de una meta o resultado deseado. (Vazquez, 2012).

Se busca dotar a un agente inteligente con:

* La representación del objetivo que se va a alcanzar.
* La representación de las acciones que puede realizar.
* La representación del entorno.
* La capacidad de generar un plan para alcanzar el objetivo.

Problema de planificación

Un problema de planificación consiste en un estado inicial el cual mediante la aplicación de una secuencia de acciones llevan a un estado final u objetivo. Se considera a la planificación como un problema de búsqueda en un espacio de estados. A continuación, se muestra un grafo que representa un problema de planeación (Ilustración 1), donde delta sub cero es el estado inicial, delta sub-n es el estado final y delta representa la acción que se ejecuta para pasar de un estado (nodo) a otro (Agentes planificadores., s. f.).

A diagram of a network

Description automatically generated

Ilustración 1 Representación gráfica de un problema de planificación

Tipos de planes

Existen dos tipos de planes: los planes parcialmente ordenados y los totalmente ordenados. Los planes parcialmente ordenados consisten en un conjunto de acciones ordenadas en las cuales se tienen restricciones en cuanto a su secuencia de ejecución, donde pueden haber más de dos ramificaciones que son ejecutadas dependiendo de la opción que se elija. Los planes totalmente ordenados consisten en acciones las cuales se van ejecutando una tras otra siguiendo una secuencia lineal (Agentes planificadores., s. f.).

Algoritmo de planificación

Un algoritmo de planificación enlista todos los estados intermedios que llevan del estado inicial al final, reduciendo su complejidad descomponiendo el problema en subobjetivos (Agentes planificadores., s. f.).

Planificación clásica

Se emplea la planificación clásica donde los entornos son:

* Completamente observables para el planificador, este percibe todo el estado, sus propiedades y el efecto de las acciones.
* Son deterministas porque se puede predecir y predefinir los efectos de las acciones que se ejecutan.
* Son finitos sus estados, acciones y efectos.
* Son estáticos, ya que cambian solo cuando el planificador actúa sobre él.
* Son discretos, es decir, son contables tanto sus acciones, objetos y efectos.

PDDL

Para programar estos planificadores se emplea el lenguaje PDDL (Planning Domain Definition Language), estándar usado para la representación de tareas de planificación clásica usado desde 1998 por el ICAPS (International Conference on Automated Planning and Scheduling). Se inspiró en UCPOP, el planificador desarrollado por la Universidad de Washington. (Berzal).

International Planning Competition

Esta competición se ha llevado a cabo desde 1998 en varias partes del mundo por el ICAPS (Competitions – ICAPS, s. f.), presenta varias pistas o tracks casi cada año, las cuales son:

* Deterministic.
* Probabilistic.
* Uncertainty.
* Learning.
* Probabilistic Continuous.
* Probabilistic Discrete.
* Temporal.
* Classical.
* Numeric.
* HTN

Estas pistas cuentan con sub categóricas en las que se registran varios equipos los cuales deben darle solución a una de varias tareas planteadas usando planificadores, estos planificadores de cumplir con una serie de condiciones propias de dicha subcategoría como el tiempo de ejecución, el límite de memoria usado, el CPU usado, el número de tareas resuelto entre otros (Competitions – ICAPS, s. f.). Por ejemplo, algunas de las tareas que se han usado en esta competición son:

* El juego de cartas Agrícola.
* Ciberseguridad.
* Procesamiento y envío de datos en una red.
* El puzzle Nurikabe.
* El juego Snake.
* El juego Spider.
* Laberinto.
* Cubo Rubik.
* Síntesis de diseño de círculos cuánticos.

Desarrollo

Para realizar esta actividad, primero se hizo la instalación de Singularity en dos computadoras, una computadora con sistema operativo Linux y otra en una máquina virtual con Ubuntu v22.04.3 (Ilustración 2).

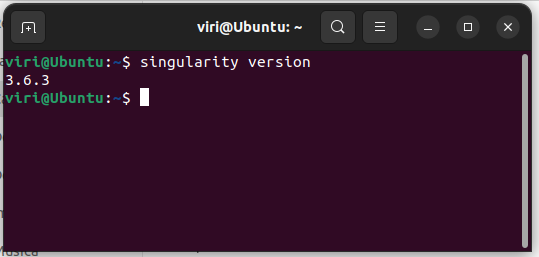


Ilustración 2 Singularity versión

* Pruebas de ejecución del planner ganador y los archivos domain y problem de la Tarea Snake

Una vez instalado Singularity se tomó el repositorio del equipo 23 Delfi1 participante y ganador en la IPC 2018 (International Planning Competition 2018) para la generación del archivo Singularity (IPC 2018, 2018). Se ejecutó en una terminal la línea de comando *wget* [*https://bitbucket.org/ipc2018-classical/team23/raw/ipc-2018-seq-opt/Singularity*](https://bitbucket.org/ipc2018-classical/team23/raw/ipc-2018-seq-opt/Singularity) (Ilustración 3).

A screenshot of a computer

Description automatically generated

Ilustración 3 Generación del archivo Singularity a partir del repositorio

del equipo 23 Delfi1

Luego en la misma terminal, con el comando *sudo**singularity build planner.img Singularity* se creó la imagen del planificador del repositorio del equipo 23 Delfi1 (Ilustración 4).

A screenshot of a computer

Description automatically generated

Ilustración 4 Se crea el archivo planner.img

La imagen del planificador se creó en la misma carpeta que el archivo Singularity (Ilustración 5), esto es importante ya que ambos archivos deben de estar juntos, de lo contrario al ejecutar los siguientes comandos se va a producir un error.

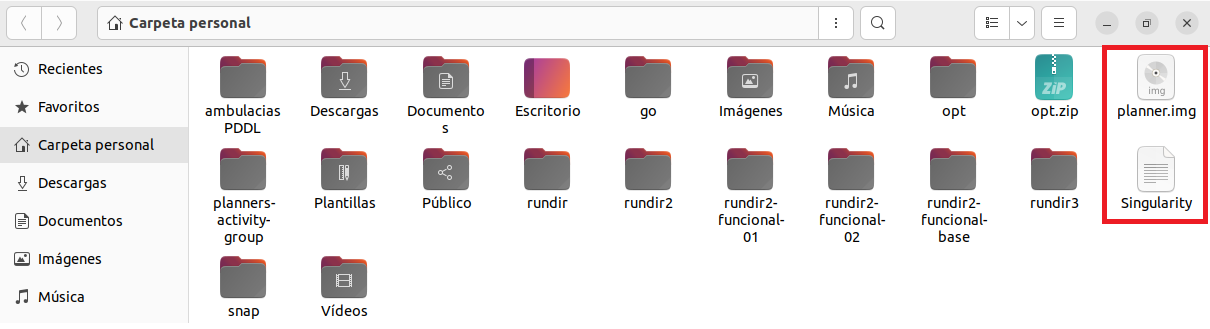


Ilustración 5 Ubicación de los archivos Planner.img y Singularity

Después se toman el archivo domain y uno de los archivos problema de la tarea Snake, se agregan a la carpeta rundir y se ejecutan los siguientes comandos (Ilustración 6):

* + RUNDIR="$(pwd)/rundir": se toma la ubicación de la carpeta que contiene los archivos domain y problem, y se debe de especificar el nombre de dicha carpeta, en este caso la carpeta se llama “rundir”.
  + DOMAIN="$RUNDIR/domain.pddl": se especifica el nombre del archivo del dominio, en este caso se llama domain.
  + PROBLEM="$RUNDIR/problem.pddl": se especifica el nombre del archivo del problema, en este caso se llama problem.
  + PLANFILE="$RUNDIR/sas\_plan": Se especifica la ubicación en donde se va a generar el archivo que contendrá el plan generado, en este caso el archivo recibirá el nombre sas\_plan y se guardará en la misma carpeta que aloja los archivos del dominio y del problema.
  + ulimit -t 1800: se define el tiempo límite.
  + ulimit -v 8388608: se define el volumen límite.
  + singularity run -C -H $RUNDIR planner.img $DOMAIN $PROBLEM $PLANFILE $COSTBOUND: este comando ejecuta el archivo domain y problem usando el archivo Singularity y planner.

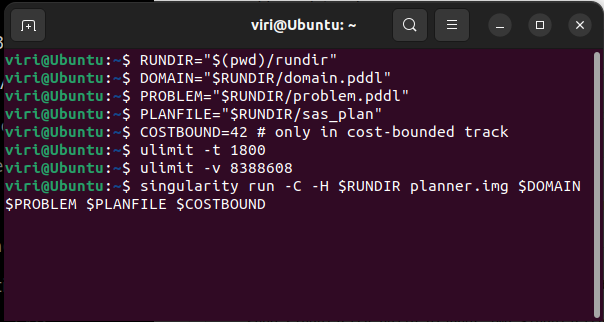


Ilustración 6 Comandos para la ejecución del domain y problem de snake en una máquina virtual

A continuación, se muestra la ejecución de los archivos domain y problem de la tarea Snake (Ilustración 7):

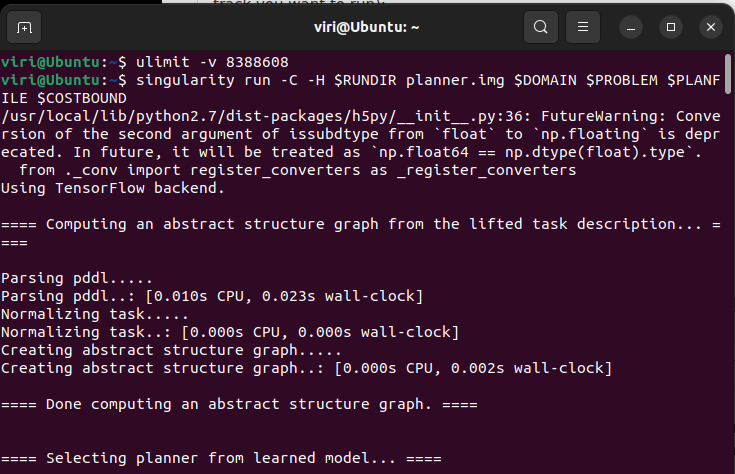


Ilustración 7 Ejecución del domain y problem de snake

A screenshot of a computer program

Description automatically generated

Ilustración 8 Finalización de ejecución de domain y problem de snake

Una vez terminada la ejecución se genera el plan de la tarea y es guardado en el archivo sas\_plan, el cual es almacenado en la carpeta “rundir”, a continuación, se muestra el plan generado para la tarea Snake (Ilustración 9).

A screenshot of a computer program

Description automatically generated

Ilustración 9 Se muestra el planner creado tras la ejecución del script de Singularity

* Descripción de escenario original de la actividad

Una vez realizada la ejecución del planificador se procede a trabajar con el siguiente problema planteado:

Se tiene un robot tipo *rover* que previamente realizó la excavación de dos rocas localizadas en la localidad 1 y localidad 2. Debido al mal tiempo, no fue posible trasladar las rocas para su análisis. Es por ello por lo que se solicita generar el plan que debe seguir el robot para llevar los minerales al laboratorio de análisis.

Debido al terreno, hay restricciones en la trayectoria: de la localidad 3 a la 1, el camino está libre y existe bidireccionalidad; de la localidad 3 a la 2, el camino solo es de una dirección; de la localidad 2 a la 4 es solo una dirección; de la localidad 3 a 4 y de 4 a 5, el camino es bidireccional. El robot solo puede cargar un mineral a la vez.

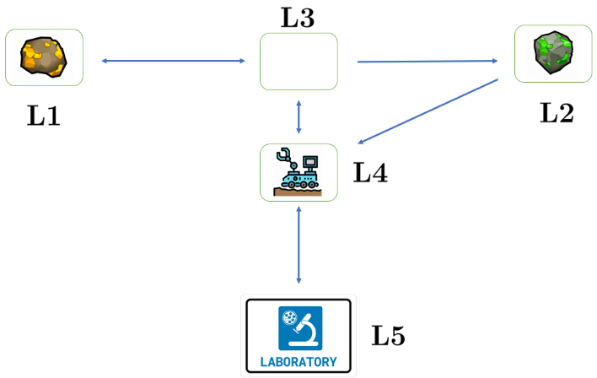


Ilustración 10 Esquema del problema

Dificultades encontradas

Conclusiones

Referencias Bibliográficas

* Asale, R.-. (s. f.). Planificación | Diccionario de la Lengua Española. «Diccionario de la lengua española» - Edición del Tricentenario. <https://dle.rae.es/planificaci%C3%B3n>
* Vazquez, J. (2012). Obtenido de <https://www.cs.upc.edu/~jvazquez/teaching/iag/transpas/4-PL1-IntroPlan>
* Berzal, F. (s.f.). *DECSAI.* Obtenido de Departamento de Ciencias de la Computación e IA: <https://elvex.ugr.es/decsai/iaio/slides/P9%20PDDL.pdf>
* IPC 2018. (2018). INTERNATIONAL PLANNING COMPETITION 2018. Recuperado 22 de agosto de 2023, de <https://ipc2018-classical.bitbucket.io/>
* Agentes planificadores. (s. f.). [Diapositivas]. Universidad Politécnica de Cataluña. <https://www.cs.upc.edu/~jvazquez/teaching/iag/transpas/4-PL1-IntroPlanificaci%c3%b3n.pdf>
* Competitions – ICAPS. (s. f.). <https://www.icaps-conference.org/competitions/>